



Marketplace Mircoservice

by Daniel Ronaldo Pangestu



PHINTRACO GROUP

Content

01 Requirement

02 Limitasi Project

03 Project Architecture

04 Tech Stack

05 Database Schema

06 Endpoint

07 Flowchart Program

08 Demo App

01 Requirement

1. Service yang diminta: Product service, Balance & Transaction service, Order service, Orchestration service
2. Penggunaan Messaging
3. Komunikasi antar service melalui web client

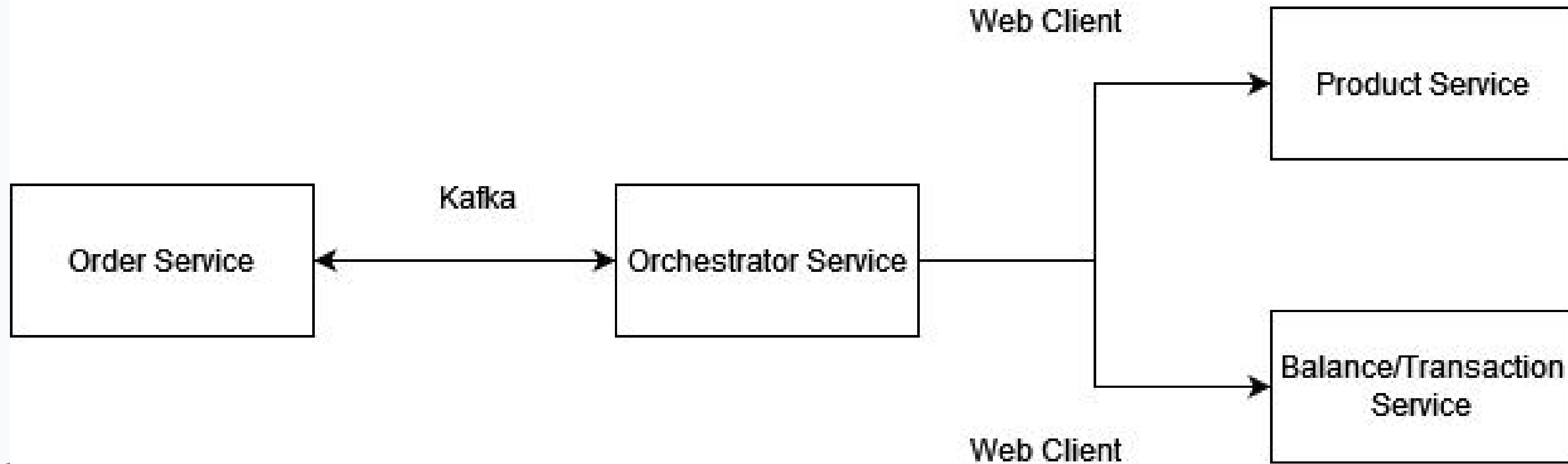
02 Limitasi Project

1. Data yang diinputkan berupa data dummy
2. Komunikasi menggunakan messaging hanya dibataskan pada order service
3. Komunikasi dengan service lain selain order service menggunakan web client
4. Fokus utama project adalah komunikasi antar service

03 Project Architecture

1. Project menggunakan arsitektur microservice
2. Microservice menggunakan Saga Orchestration Pattern








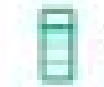

03 Project Architecture



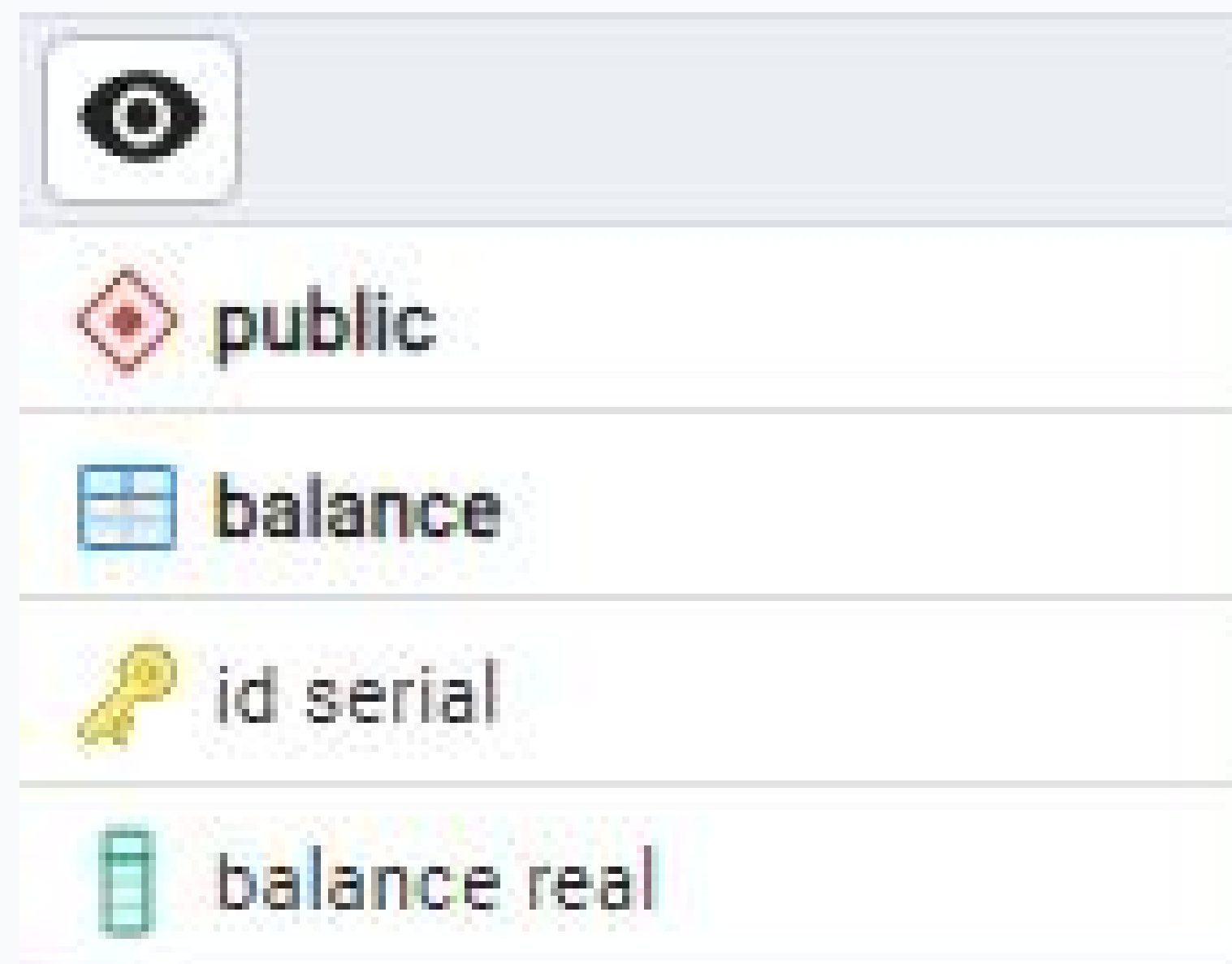
04 Tech Stack

1. Java Language
2. Spring Boot WebFlux
3. Kafka
4. PostgreSQL
5. Spring Boot R2DBC

05 Database Schema (Product)


 public
 product
 id serial
 name character varying(255)
 price real
 category character varying(255)
 created_at timestamp without time zone
 description character varying(255)
 image_url character varying(255)
 stock_quantity integer
 updated_at timestamp without time zone



05 Database Schema (Balance)



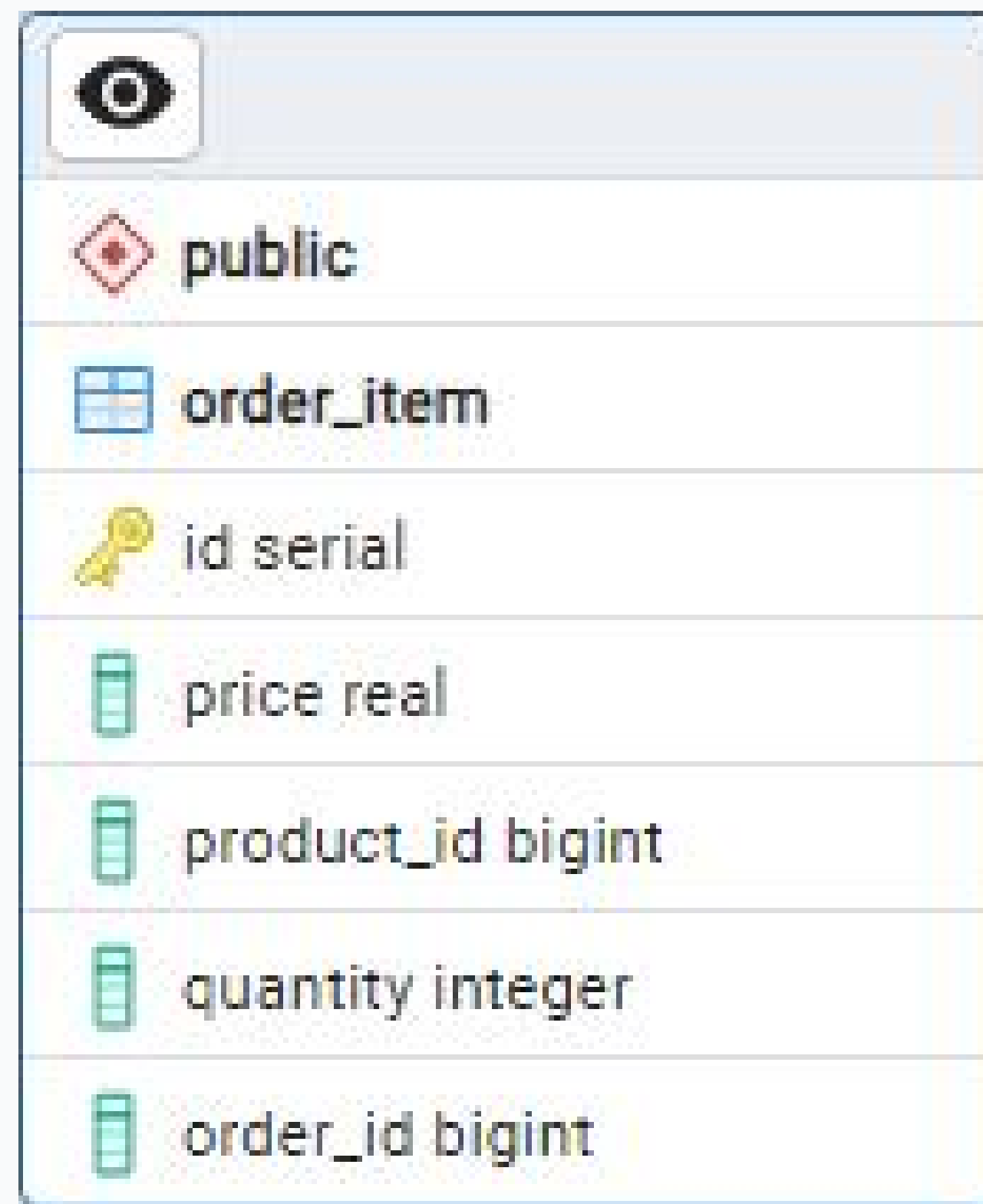
05 Database Schema (Transaction)

public
transaction
id serial
amount real
order_id bigint
payment_date timestamp without time zone
mode character varying(255)
status character varying(255)
reference_number character varying(255)

05 Database Schema (Order)

	
	public
	orders
	id serial
	billing_address character varying(255)
	customer_id bigint
	order_date timestamp without time zone
	order_status character varying(255)
	payment_method character varying(255)
	shipping_address character varying(255)
	total_amount real

05 Database Schema (Order Item)



The image shows a database schema diagram for the 'order_item' table. It is presented in a window-like interface with a toolbar at the top containing an eye icon. The schema is organized into a tree structure. The 'public' schema is expanded, showing the 'order_item' table. The table's columns are listed below it, each with a data type and a corresponding icon: a key icon for the primary key 'id' (serial), and a document icon for 'price' (real), 'product_id' (bigint), 'quantity' (integer), and 'order_id' (bigint).

public
order_item
id serial
price real
product_id bigint
quantity integer
order_id bigint

06 Endpoint (Product)

Servers		
http://localhost:8081 - Generated server url		
GET	/api/product/{id}	Get Product by ID
PUT	/api/product/{id}	Update Product by ID
DELETE	/api/product/{id}	Delete Product by ID
GET	/api/product	Get All Products
POST	/api/product	Create a New Product
DELETE	/api/product	Delete All Products
PATCH	/api/product/{id}/deduct	Deduct Product Stock by ID
PATCH	/api/product/{id}/addStock	Add Stock to Product by ID
GET	/api/product/	Get Products by Name

06 Endpoint (Balance)

Servers

http://localhost:8083 - Generated server url

GET

/api/balance/{id} Get Balance by Id

PUT

/api/balance/{id} Update Balance by Id

DELETE

/api/balance/{id} Delete Balance by Id

GET

/api/balance Get All Balances

POST

/api/balance Create a new Balance

DELETE

/api/balance Delete All Balances

06 Endpoint (Transaction)

Servers

`http://localhost:8083 - Generated server url` ▼

GET	<code>/api/transaction/{id}</code>	Get Transaction by Id
PUT	<code>/api/transaction/{id}</code>	Update Transaction by Id
DELETE	<code>/api/transaction/{id}</code>	Delete Transaction by Id
GET	<code>/api/transaction</code>	Get All Transactions
POST	<code>/api/transaction</code>	Create a new Transaction
DELETE	<code>/api/transaction</code>	Delete All Transactions

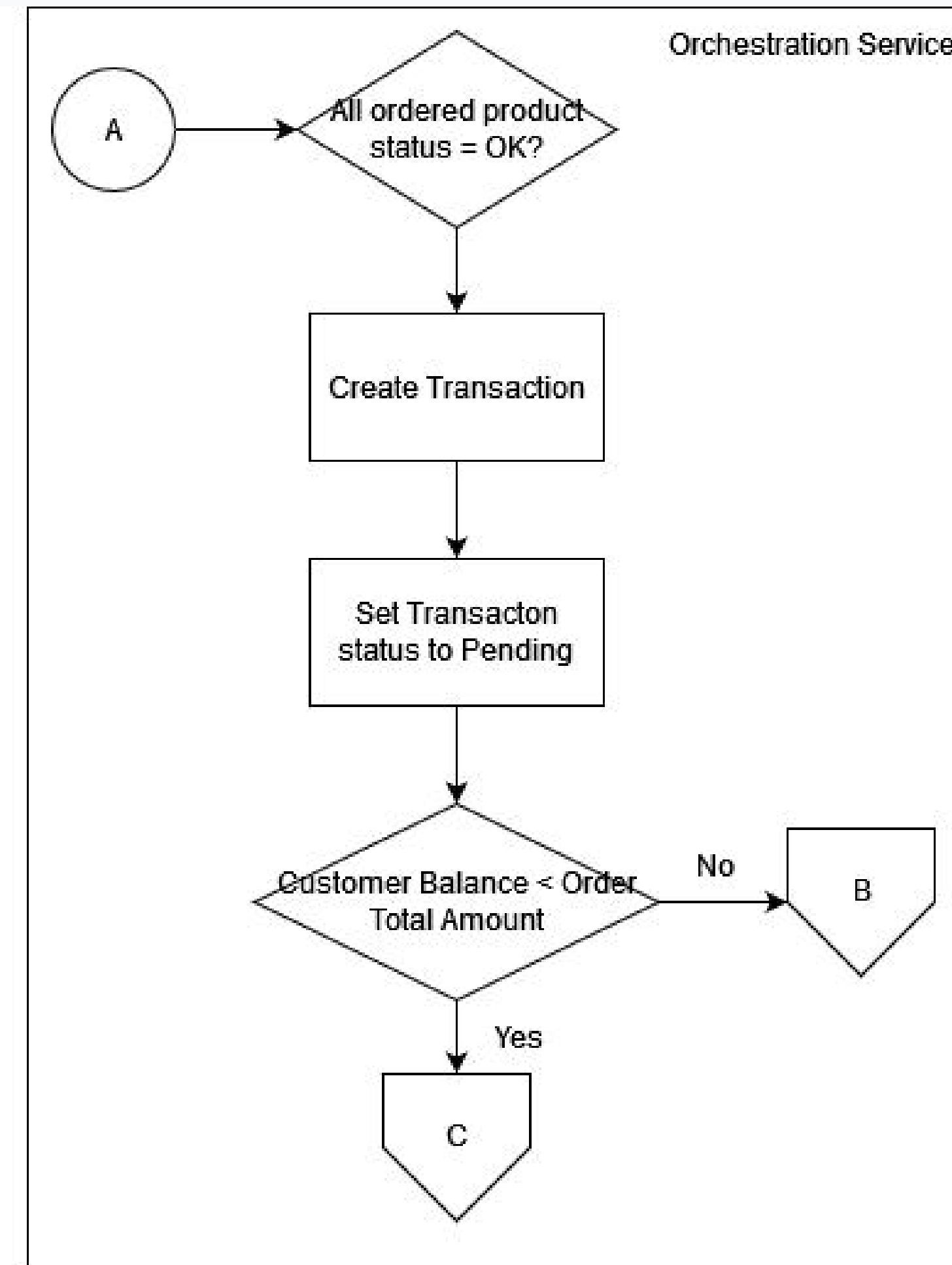
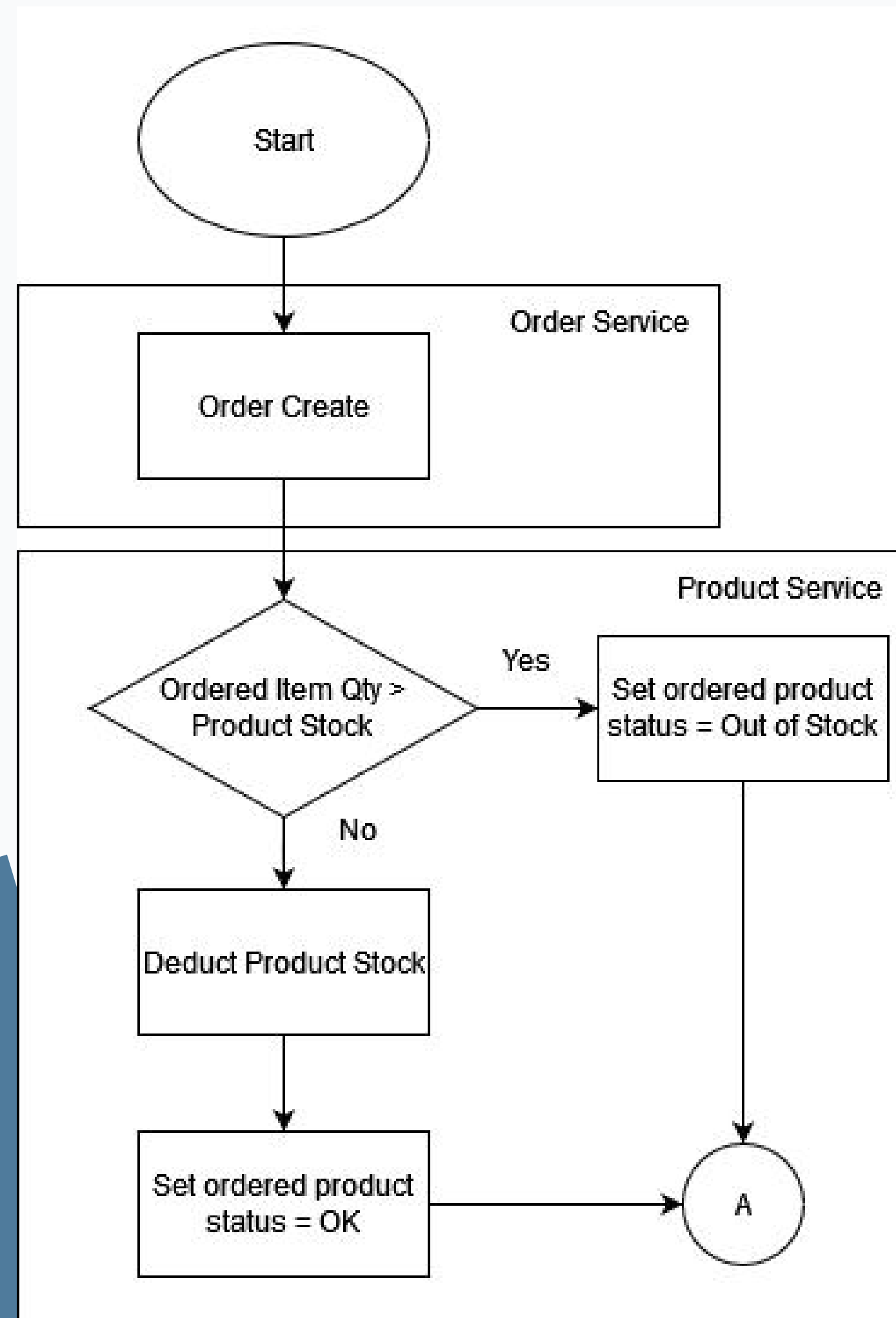
06 Endpoint (Order)

Servers		
http://localhost:8084 - Generated server url		
GET	/api/order/{id}	Get Order by Id
PUT	/api/order/{id}	Update Order by Id
DELETE	/api/order/{id}	Delete Order by Id
GET	/api/order	Get All Orders
POST	/api/order	Create a new Order
DELETE	/api/order	Delete All Orders
PATCH	/api/order/{id}/updateStatus	Update Order Status by Id

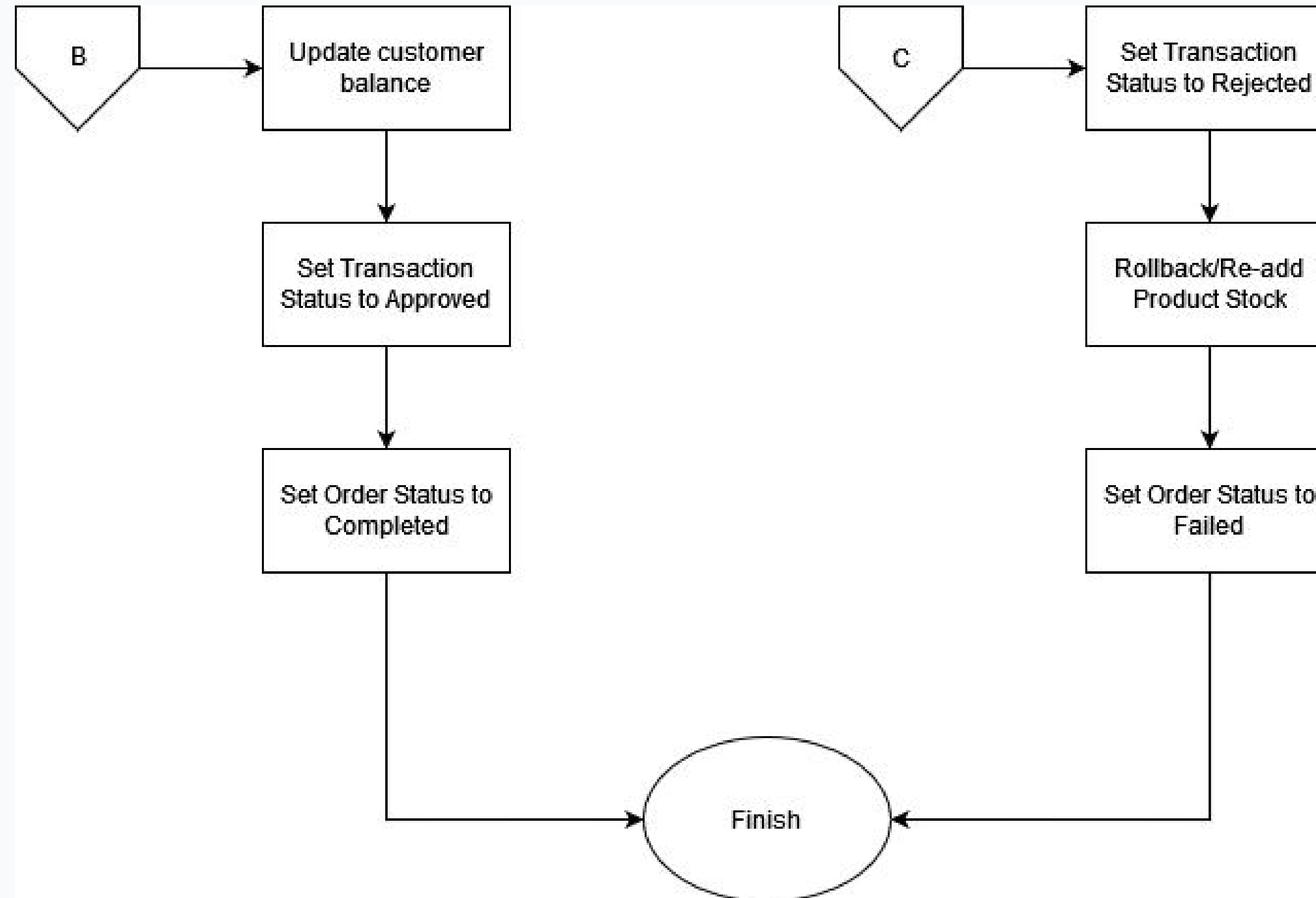
06 Endpoint (Order Item)

Servers		
http://localhost:8084 - Generated server url		
GET	/api/order-item/{id}	Get Order Item by Id
PUT	/api/order-item/{id}	Update Order Item by Id
DELETE	/api/order-item/{id}	Delete Order Item by Id
GET	/api/order-item	Get All Order Items
POST	/api/order-item	Create a new Order Item
DELETE	/api/order-item	Delete All Order Items

07 Flowchart Program (Part 1)



07 Flowchart Program (Part 2)



08 Demo App



Thank You